RECOGNITION OF TRACK DEFECTS THROUGH MEASURED ACCELERATION USING A RECURRENT NEURAL NETWORK

SEBASTIAN BAHAMON-BLANCO, SEBASTIAN RAPP, YI ZHANG, JING LIU & ULLRICH MARTIN
Institute of Railway and Transportation Engineering, University of Stuttgart, Germany.

ABSTRACT

As part of an optimized maintenance strategy, track monitoring should provide information to predict track faults at an early stage. This is possible by continuously measuring the axle box accelerations and using artificial intelligence, which can detect short wave defects on the railway track with high accuracy. Such short wave defects include rail breaks, cracks, and local irregularities (mud spots). These types of faults can reduce the track quality in a short period of time.

Different track irregularities were simulated in a track-vehicle scale model to generate acceleration data for typical track defects. The main focus of the current research is on recognition of local irregularities in the track-vehicle scale model. To implement the artificial intelligence, a Recurrent Neural Network is used to show the procedure and the results of recognition of track defects. The architecture and components of the neural network used are described in detail in this article. At the end of the article, a table summarizing the results of the different models trained for detecting the local irregularities in the track-vehicle scale model is presented.

Keywords: artificial intelligence, deep learning, detection, local instability, maintenance, railway.

1 INTRODUCTION

Due to train traffic and climate factors, the ballast track system is permanently affected. Hence, this can lead to formation of local irregularities (mud spots) with reduced track stiffness and degradation of the track geometry. Local irregularities can decrease the track quality in a short period of time, inducing several negative effects such as accumulation of excess pore water pressure within the substructure, rising of the underlying fine-grained soil under traffic load (mud pumping effect), contamination of the ballast bed, reduction of subgrade resistance, and degradation of track geometry in a longitudinal level (even derailments, in the worst case).

Artificial intelligence is an approach that is becoming very important in different sciences, including medicine, finance, security, and also, engineering. One of its branches is known as machine learning, which uses algorithms such as support vector machine, generalized matrix learning vector quantization or decision trees, and, for the other side, artificial neural network (ANN).

Detection of local irregularities (mud spots) by signal processing methods was studied in [1]. The following methods were investigated: analyses of typical amplitude ranges, a frequency analysis (windowed spectral acceleration density), cross correlation, wavelet analysis, and fault detection based on the classification of peaks by amplitude and wavelength. All these methods successfully detected the local irregularities at correct positions in the track-vehicle scale model [1].

Additionally, the recognition of track faults in the track-vehicle scale model was investigated in [2] by bagged decision tree algorithms. These simple machine-learning algorithms can automatically classify the vertical acceleration signal in three classes such as noise, joint/cracks, and local irregularities (mud spots). In total, three models A, B, and C have been created. These models have achieved between 84% and 88% accuracy in local irregularity detection; but to increase this accuracy, this research focuses on a special type of ANN (model D) for pattern recognition.

ISSN: 2046-0546 (paper format), ISSN: 2046-0554 (online), http://www.witpress.com/journals
DOI: 10.2495/CMEM-V8-N3-270-280

2 TRACK-VEHICLE SCALE MODEL

An oval track-vehicle scale model of 4.04 m track length (scale 1:87) simulates an elevated railway (see Fig. 1). This model allows to change the elevations of the track due to its spring screw construction and add track faults with different amplitudes, wavelengths, and twists. The knowledge about fault types and their characteristics in the real system is still lacking. However, a track-vehicle scale model represents a big advantage to train models for pattern recognition because the faults can be easily set and their exact positions and the acceleration signals are well known [1]. In addition, the physical similarity between the track-vehicle scale model and the real system was analyzed by dimensional analyses in [1]. The acceleration generated with the track-vehicle scale model was comparable to values for the real railway system in literature (shape and maximum acceleration). Moreover, the vertical acceleration is similar to the real system at the track section with a local irregularity. The track-vehicle scale model contains four rail joints, two rail cracks, one local irregularity (mud spot) [3], and the entrance/exit to the bridge. The position of each fault is marked in Fig. 1.

2.1 Simulated track irregularities

The classification of the fault types depends on the wavelength in a longitudinal level. In this model, the following faults can be detected:

- rail defects such as rail cracks and joints and
- track irregularities with a wavelength between 1 and 25 m, such as cracked sleepers, defect fastener, hanging sleepers, and local irregularities (mud spots).

The fault types in the track-vehicle scale model and in the real system are shown in [2]. The vehicle continuously gains acceleration, and when it passes a fault, the sensor [5] detects an increment of the vertical acceleration. The generated signal is used to recognize and classify different fault types. Every fault type has a standard waveform with specific characteristics [1]. For example, rail cracks and joints generate high amplitudes, short wavelengths, and two peaks (vehicle excitation caused by the two axes of the first car). On the other hand, the entrance and exit to the bridge and the local irregularity (mud spot) generate accelerations with low amplitudes and large wavelengths.

2.2 Local measurement system

The measurement system in the track-vehicle scale model contains one locomotive and two cars. The first car carries the battery and the second car has the sensor module which measures the acceleration data with a sample rate of 500 Hz. To increase the natural frequency of

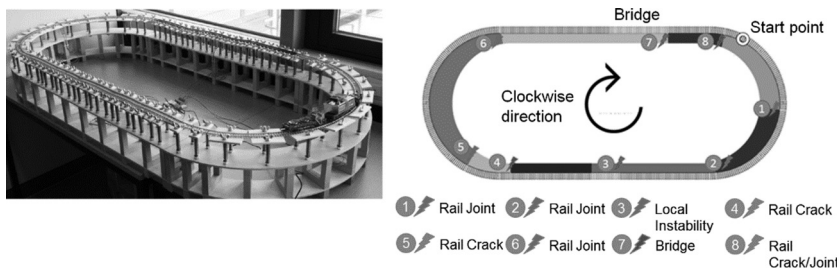

Figure 1: Track-vehicle scale model and position of the track defects [3, 4].

the measuring vehicle and to generate a pumping effect at the track irregularity (mud spot), extra weight is placed on the car with the sensor module. The vehicle speed is roughly 0.37 m/s while passing the track faults. After finishing each lap, the train always stops at the same position for about 3 s until another lap starts (Fig. 1, start point) [3]. Due to the absence of a spring and suspension system at the vehicle, there are no external influences which affect the measured acceleration data. For this reason, the collected data is equivalent to the axle box acceleration on a real train [1].

3 ARTIFICIAL NEURAL NETWORK

An ANN is a popular machine learning method which simulates the networked structure of neurons in the brain, with layers of connected neurons. It is especially suitable for modeling high-level abstractions in datasets that can be trained to recognize patterns, classify data, and forecast future events [6]. In this research, model D (created by using ANN) is explained.

To develop new methods for detecting faults in the track-vehicle scale model while using more data, it is important to apply Recurrent Neural Network (RNN), which is usually used for pattern recognition.

3.1 Recurrent Neural Network

RNN is designed for sequential data like text sentences or time series [7] where the previous outputs of hidden neurons are used to generate the response for the current input using a loop structure. An important point about sequences is that successive data are interdependent. Therefore, it is helpful to receive a particular input only after the earlier inputs have already been received and converted into a hidden state [7]. The RNN allows the input to interact directly with the hidden state created from the inputs at previous time stamps [7].

3.1.1 Long short-term memory

Long short-term memory (LSTM) is a type of RNN which reuses the output from a previous step as an input for the next step. Like all ANN, the neuron performs a calculation using the inputs and returns an output value in an RNN. This output is then used along with the next element as the inputs for the next step and so on [8]. LSTM is designed for applications where the input is an ordered sequence in which information from earlier in the sequence may be important.

3.2 Architecture

The architecture in this research is similar to the one used by [9]; but instead of having only six laps as the input, the current network contains several laps as input, and in order to perform a non-linear classification for the training data, the ANN contains two LSTM layers instead of just one: to avoid overfitting, a dropout layer is added – this layer has the effect of simulating a large number of networks with different ANN structures and, as a result, it makes neurons in the ANN generally more robust to the inputs [10].

The input data represents sequential measured acceleration in a vertical direction, which is around 640.000 data points. This input is separated into 10 groups (otherwise, it overloads the neural network), which will be parallelly fed into the ANN [11].

3.2.1 Hidden layers

A hidden layer in an ANN is a layer in between the input and the output layers, where artificial neurons take in a set of weighted inputs and produce an output through an activation

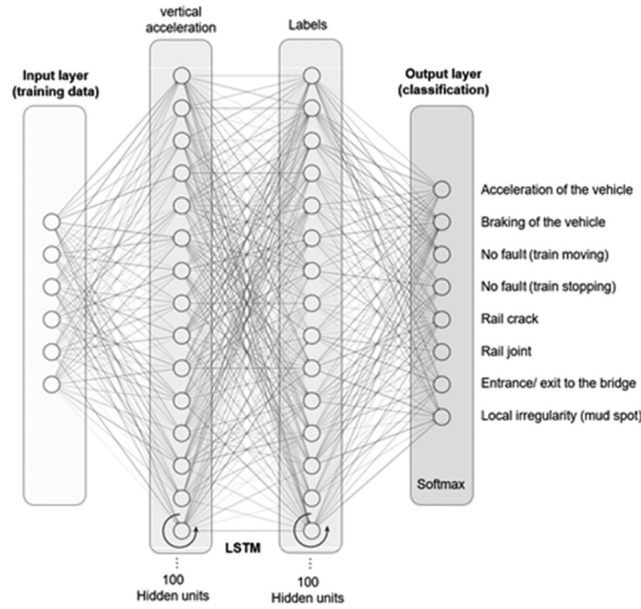

Figure 2: Architecture of the RNN [13].

function. These layers are called 'hidden' because their values are not given in the data; instead, the model must determine which concepts are useful for explaining the relationships in the observed data [12]. For this case, two layers are used: one for feeding the vertical accelerations and one for feeding the labels. The number of units depends on the complexity of the classification problem. In this case, 100 units are selected from the experience [11].

Finally, eight classes are specified by including a fully connected layer. The number of output layers is the number of classes the training data should be divided into.

3.2.2 Activation layer

The activation function defines the output of a neuron in terms of a local induced field. It is described as [1]:

$$x \rightarrow a = f\left(w * x + b\right) \tag{1}$$

where f is the activated function which works with synaptic weights (w) that refer to the strength of the relative influence of a connection between two neurons which are changed by using a learning rule. The input (x) here is the vertical acceleration. Also, the bias (b) controls how the neuron is predisposed to trigger a 1 or 0 independent of the weights. A high bias causes the neuron to require a higher input to generate an output of 1. A low bias makes it easier.

3.2.3 Dropout layer

Dropout is used to improve overfitting on ANN, deactivating randomly a specific percentage of the neurons, which become more insensitive to other weights, making the model more robust. Figure 3 shows how the synaptic connection of some neurons is deactivated and the relation between the impact of this technique and the error rate. If the training error rate is smaller than the testing error rate, the model is going too deep. This can be unnecessary and

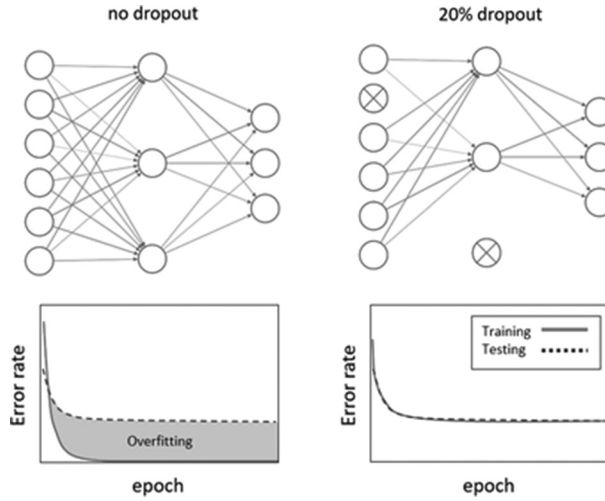

Figure 3: Deactivation of neurons and optimization of the model by using dropout.

consumes time and memory. The model is optimized when the training and testing error rates are nearly the same, because it is achieving the main objective and it is not doing additional tasks [14]. But a high dropout creates an underfitted model, and to avoid this, 20% of dropout is the recommended value to this layer [15].

3.2.4 Output layer

The Softmax function is generally used in the output layer. It converts every output to be in the range of 0 and 1, just like the sigmoid function. But it divides each output, so that the total sum of the outputs gives the probability that a certain class is correctly identified [16].

3.2.5 Hyper-parameter adjustment

This research uses the adaptive moment estimation (ADAM) solver, which performs better with RNNs like LSTMs than the default stochastic gradient descent with momentum (SGDM) solver [11]. In addition to the layers, some adjustments should be set. The parameters are shown in Table 1.

3.3 Labeling

This process is described in detail in [2]. Every sample number has a numerical label from 1 to 8 (see Table 2), which is assigned by executing an automatic classification process. A moving average filter smoothens out the noisy data of the vertical acceleration signal. With the calculated absolute values of the vertical accelerations, peaks of the simulated track irregularities are detected automatically when the acceleration is higher than a defined acceleration threshold and a set distance. After a dataset with the peak position, the maximum acceleration and the sample distance between two consecutive peaks are generated. A specific sample range is then designated from the unified peak position. The automatic labeling process works only for the generated training data. Each lap shows a large similarity and a kind of periodicity. If the starting point or the direction of vehicle movement is changed, use of the automatic classification process is no longer possible [2].

Table 1: Description of the hyper-parameter adjustments [11].

Parameters	Description	Value
MaxEpochs	It allows the network to make several passes through the training data	20
MiniBatchSize	It directs the network to calculate a specific training signal at a time	120
InitialLearnRate	It helps speed up the training process; in the case of 0.001, the network learns too slowly and the loss is higher	0.01
SequenceLength	It breaks the signal into smaller pieces, so that the machine does not run out of memory by looking at too much data at one time	500
GradientThreshold	It stabilizes the training process by preventing gradients from getting too large	1
Plots	It generates plots that show a graphic of the training progress as the number of iterations increases	'Training-progress'
Verbose	It suppresses the table output that corresponds to the data shown in the plot	false

3.4 Training data

There are pre-existing trained data which are obtained by doing six tests using the highest speed (0.37 m/s) and stopping momentarily after the eighth fault (see Fig. 1). In total, there are 61 laps with the same sequence of faults; but to train the RNN, the original signal is modified to make the process more effective. Therefore, the starting point of each lap is randomly selected in order to make the fault sequence partially disappear. This is repeated three times to extract 183 laps from the original 61 laps. Additionally, the full original signal is mirrored to simulate a change of direction and the starting points of each lap are randomly selected three times. In total, the training data has 366 labeled laps without sequences with several starting points and in both directions.

It is important to mention that the collected data is not filtered as part of the future work of.

3.5 Testing data

The RNN is tested with 14 tests, with 6 of them having the same configuration as the training data and 8 with some basic changes in the starting point, speed, stopping process, and direction. To use these tests like training data, each sample number is labeled from 1 to 8 according to a specific class as it is shown in Table 3. These 14 tests were also used to test the simple machine learning models A, B, and C.

4 DETECTION OF THE LOCAL IRREGULARITY (MUD SPOT)

The RNN (model D) achieves good results. The plots show that all the 61 laps are classified into the eight classes presented in Table 3, as shown in Fig. 4. This is a bulleted list.

Table 2: Description of the labeled acceleration data for the models A, B, C, and D.

Bagged Decision Trees Model			Recurrent Neural Network Model		Fault numbered in Fig. 1
Label	A B C Class		Label	D Class	
1	Noise		1	Acceleration of the vehicle	---
			2	Braking of the vehicle	---
			3	No fault (train moving)	---
			4	No fault (train stopping)	---
2	Joint, crack		5	Rail crack	4 and 8
			6	Rail joint	1, 2, 5, and 6
			7	Entrance/exit to the bridge	7
3	Local instability		8	Local irregularity (mud spot)	3

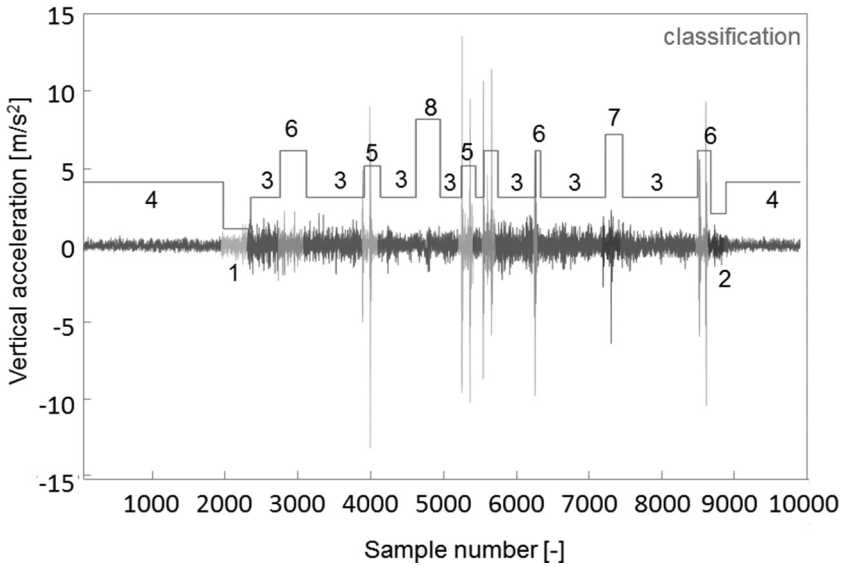

Figure 4: Detection of different classes by RNN.

4.1 Comparison of model D with the simple machine learning models A, B, and C (Bagged Decision Trees)

In most of the cases, the predictions are right; but there are some specific cases where the predictions fail. Figure 5 shows that the results of RNN (model D) are better in tests 1–6 than the models A, B, and C (Bagged Decision Trees), and even when the average accuracy of model D is lower, it is close to the accuracy of the other models and it is able to classify not

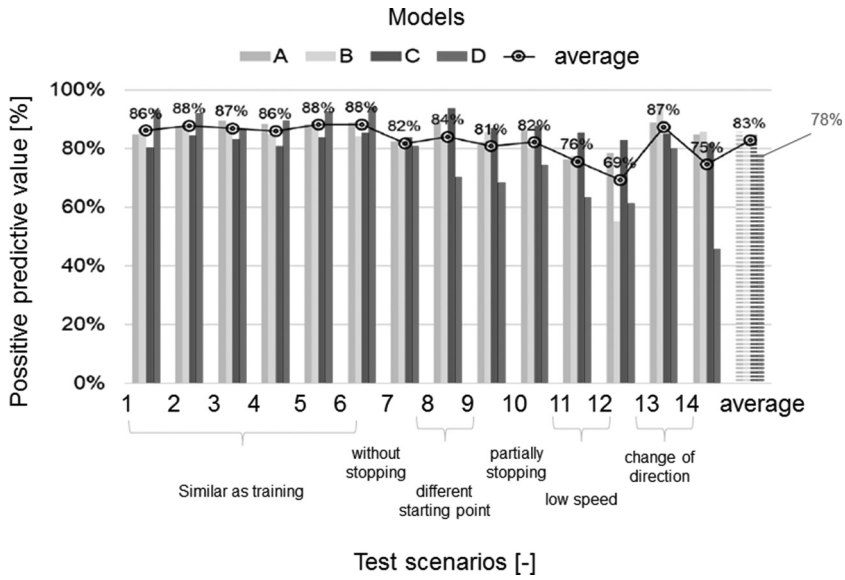

Figure 5: Comparison of the accuracy of each trained model for the tests 1–14.

just three classes, but eight, as shown in Table 2. Also, while the simple machine learning models achieve an accuracy of local irregularities detection between 84% and 88% [10] and model D (RNN) achieves 97%. The accuracy of the models is calculated based on the correct predictions divided by the sum of the actual number of local irregularities and the wrong detections:

$$Positive\,Predictive\,Value = \frac{Correct}{\left(Expected + Wrong\right)} * 100 \quad [\%] \tag{2}$$

The most challenging tests to classify correctly, according to the machine learning models, are the tests 7, 9, 11, 12, and 14 because they have a specific configuration that differs somehow from the training data. Nevertheless, the RNN can classify almost all the tests without wrong predictions, except test 14 which is running in a different direction.

5 CLASSES RECOGNITION

The positive predicted values for model D are shown in Fig. 6. It shows that the classes with higher accuracy are the entrance/exit to the bridge, local irregularities (mud spots), rail crack, and joint. Class 1 (acceleration of the vehicle) has a good performance except when the train does not stop in every lap (test 10), when the train is moving with a slow speed (tests 11 and 12), and also when the sensor module is pulling the vehicle (test 14). Class 2 (braking of the vehicle) is always detected after the eighth fault, even when the train is not stopping in every lap (test 7), when it is not stopping in the starting point position, for example, for test 1 (tests 8 and 9), and when the sensor module is pulling the vehicle (test 14). The detection of the entrance and exit of the bridge is generally good, but in tests 13 and 14, this class 7 is wrongly detected most of the time. Class 8 (local irregularity) in sum achieves a good result. Only two incomplete predictions were done in tests 11 and 12 (with slow speed) and three wrong

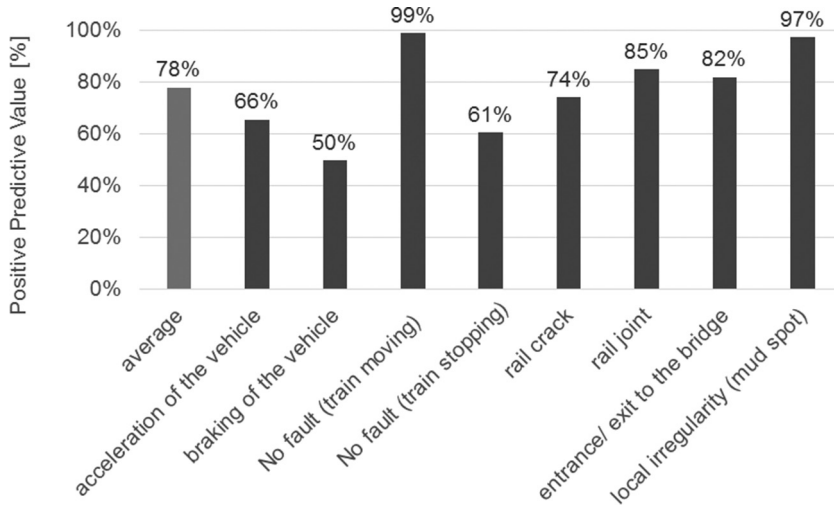

Figure 6: Class recognition of model D.

predictions in test 14. The detection of class 5 (rail crack) is always good; but in several cases, the joint is classified as a crack. Class 6 (rail joint) also achieves good results, except in tests 12 and 14 because of a slow speed and the direction of the sensor module. Class 3 (train moving) is perfectly detected, but it is an expected result because the amplitude of the signal is nearly constant. Class 4 (train stopping) achieves good results; but sometimes, the very slow speed is classified as train stopping. It is clearly observed in the test with slow speed (tests 11 and 12).

6 CONCLUSIONS

In this paper was presented a track-vehicle scale model for the simulation of typical track faults. The characteristics of a local irregularity (mud spot) were described and generated in the track-vehicle scale model. The goal was the recognition of the mud spot by continuous measured vertical acceleration in combination with RNN. Finally, the results of pattern recognition achieved by simple machine learning algorithms (Bagged Decision Trees) and RNN were compared.

The acceleration signal generated by the local irregularity (mud spot) has low amplitudes, and is therefore quite similar to the signal noise. Hence, recognition of the local irregularity (mud spot) is not always successful with signal processing methods like, for example, a normalized cross correlation or wavelet analysis, especially in cases in which the vehicle speed or moving direction changes. In contrast, the machine learning algorithms can achieve good results on alteration changed conditions, detecting the local irregularity (mud spots) in the track-vehicle scale model. With the Bagged Decision Trees method, the fault recognition is limited to three classes. In comparison, the RNN can improve the accuracy of the previous models of Bagged Decision Trees. The local irregularity of the 14 testing data is detected with 97% accuracy with the RNN (model D). Overall, this is much better than 84% and 88% achieved in previous studies with the Bagged Decision Trees method (models A, B, C) [2].

Training data of the RNN is more complex than the ones used to train the models with Bagged Decision Trees because the acceleration signal is shuffled to strengthen the training

conditions. The RNN is able to classify any sample number, independent of the conditions around.

Figure 5 shows clearly that the first six tests, with similar characteristics of the training data, present the highest accuracy. It can also be observed in Fig. 5 that the speed of the vehicle influences the acceleration data, and therefore recognition of the track defects. The worst values of the accuracy belong to the tests 7, 11, and 12. These tests have a variation in speed, which affects the amplitude of the vertical acceleration at track fault positions.

The RNN is a simple network with four layers: input, two LSTM, and output. Even when the RNN is trained with test data that differs from some of the testing data, the predictions are good. For future research, the RNN will be trained with different configurations, varying, for example, the vehicle direction, speed, the position of the sensor module, and the starting point of the measuring vehicle. The variation of the conditions will improve the model and the predictions will be more accurate. To find a better accuracy of the RNN, the settings of the layers will also be optimized in future work.

ACKNOWLEDGMENT

We thank Mrs. Elena Umanskaya for the linguistic revision of this paper.

FUNDING

The research described in this paper is part of the EPIB project which is financed by the DFG (German Research Foundation).

REFERENCES

[1] Rapp, S. Martin, U. Strähle, M. & Scheffbuch, M., Track-vehicle scale model for evaluating local track defects detection methods. *Transportation Geotechnics*, **19**, pp. 9–19, 2019. https://doi.org/10.1016/j.trgeo.2019.01.001

[2] Bahamon-Blanco, S. Rapp, S. Rupp, C. Liu, J. & Martin, U., Recognition of track defects through measured acceleration P1 & P2. *7th International Conference of EACEF (European Asian Civil Engineering Forum)*, **615**, 2019.

[3] Rapp, S. & Martin, U., Erkennung von punktuellen Unstetigkeitsstellen am Fahrweg am Beispiel eines Fahrzeug-Fahrwegmodells – Ansatz zur Modellbildung (EPIB 1.1). *Auftaktworkshop EPIB*, 2018.

[4] Bahamon, S., Detection of local instabilities on a scale vehicle-track model through measured accelerations of the vehicle. *Master Thesis, Institute of Railway and Transportation Engineering of the University of Stuttgart*, 2018.

[5] Sander, K., Anwendung Beschleunigungssensormodul Version 3.*Kurzform*, 2019.

[6] Zaccone, G., Karim, R. & Menshawy, A., Deep Learning With TensorFlow: Explore Neural Networks With Python. *Packt Publishing*, p. 72, 2017.

[7] Aggarwal, C., Neural networks and deep learning. *Springer international publishing AG*, p. 38, 2018.

[8] Deep Learning: Long Short-Term Memory Networks (LSTMs), Online. https://www.youtube.com/watch?v=5dMXyiWddYs. Accessed on: 4 May 2020.

[9] Hopkins, B., A Wavelet-Based Rail Surface Defect Prediction and Detection Algorithm. *Virginia Polytechnic Institute and State University*, pp. 74–82, 2012.

[10] How to Reduce Overfitting With Dropout Regularization in Keras, Online. https://machinelearningmastery.com/how-to-reduce-overfitting-with-dropout-regularization-in-keras. Accessed on: 10 May. 2020.

[11] Zhang, Y., Work Summary Deep Learning. *Institute of Railway and Transportation Engineering of the University of Stuttgart*, 2019.
[12] Goodfellow, I., Deep Learning. mitp Verlags GmbH & Co, p. 6, 2018.
[13] NN-SVG Publication-ready NN-architecture schematics, Online. http://alexlenail.me/NN-SVG/index.html. Accessed on: 4 May 2020.
[14] Yang, B., Class of Deep learning Chapter 5: Advanced optimizaion thechniques, Institute of signal processing and system theory. *of the University of Stuttgart*, 2018.
[15] Srivastava, N., Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, **15**, pp. 1929–1958, 2014.
[16] What is the role of the activation function in a neural network? How does this function in a human neural network system? Online. https://www.quora.com/What-is-the-role-of-the-activation-function-in-a-neural-network-How-does-this-function-in-a-human-neural-network-system2018. Accessed on: 4 May 2020.